K-Mean Class Family

Andres Mendez-Vazquez
Cinvestav Guadalajara
amendez@gdl.cinvestav.mx

Abstract

I am putting together a class in python that implement several of the k-means algorithms. This is still a work under progress so excuses any mistake.

1. Introduction

This class implements a series of classic algorithms from the k-mean family. The algorithms being implemented are:

- K-Means
- K-Medians
- K-Centers
- Fast K-Medoids
- Fuzzy C-Means

2. Auxiliary Functions

2.1. Canopy Initialization

The Centroid Initialization

2.2. KMetric

Allows to select three different metrics:

- Euclidean Metric - The only differentiable metric in the list (a.k.a. No singular points).

$$dist(\mathbf{x}, \mathbf{y}) = ||\mathbf{x} - \mathbf{y}|| = \sqrt{(\mathbf{x} - \mathbf{y})^T (\mathbf{x} - \mathbf{y})} \tag{1}$$

- Manhattan Metric

$$dist(\mathbf{x}, \mathbf{y}) = ||\mathbf{x} - \mathbf{y}||_1 = \sum_{i=1}^{n} |x_i - y_i| \tag{2}$$

- Chebyshev Metric

$$dist(\mathbf{x}, \mathbf{y}) = \max_i |x_i - y_i| \tag{3}$$

2.3. set_k

It allows to reset the number of clusters for different experiments

3. Implemented Algorithms

3.1. K-Means

This is the classic algorithm was first proposed by Stuart Lloyd in 1957 as a technique for pulse-code modulation. Yes! Signal processing.

It implements an algorithm to minimize the following cost function

$$\sum_{k=1}^{N} \sum_{i:\boldsymbol{x}_i \in C_k} \|\boldsymbol{x}_i - \boldsymbol{\mu}_k\|^2 = \sum_{k=1}^{N} \sum_{i:\boldsymbol{x}_i \in C_k} (\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T (\boldsymbol{x}_i - \boldsymbol{\mu}_k) \tag{4}$$

3.1.1. The Main Algorithm

The main algorithm is described below.

K-means(X, k)

1. Randomly choose K data points (seeds) to be the initial centroids, cluster centers,
   - $\{\mathbf{v}_1, \cdots, \mathbf{v}_k\}$
2. Assign each data point to the closest centroid
   - $c_i = \arg\min_j \{dist(\mathbf{x}_i - \mathbf{v}_j)\}$
3. Re-compute the centroids using the current cluster memberships.
   - $\mathbf{v}_j = \dfrac{\sum_{i=1}^{n} I(c_i = j)\mathbf{x}_i}{\sum_{i=1}^{n} I(c_i = j)}$
4. If a convergence criterion is not met, go to 2.

3.2. K-Medians

The k-Median algorithm is a concave minimization with cost function

$$\min_{C,D} \sum_{i=1}^{m} \min_{l=1,\ldots,k} \|D_{il}\|_p \, s.t \; - D_{il} \leq A^T - C_l \leq D_{il}, i = 1, ..., m, l = 1, ..., k \tag{5}$$

3.3. K-Centers

The K-center criterion partitions the points into k clusters so as to minimize the maximum distance of any point to its cluster center. It minimizes the worst case distance to the centroids

$$\min_{S} \max_{k=1,\ldots,K} \max_{i:x_i \in C_k} (\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T (\boldsymbol{x}_i - \boldsymbol{\mu}_k) \tag{6}$$

K-centers Algorithm

- Step 1
  - Randomly select an object $\boldsymbol{x}_j$ from S, let $\boldsymbol{h}_1 = \boldsymbol{x}_j,\ H = \{\boldsymbol{h}_1\}$.
  - It does not matter how x_j is selected.
- Step 2
  - For $j = 1$ to n:
    * $dist\left(\boldsymbol{x}_i\right) = L\left(\boldsymbol{x}_i, \boldsymbol{h}_1\right).$
    * $cluster\left(\boldsymbol{x}_i\right)$
- Step 3
  - For $i = 2$ to k
    1. $D = \max_{\boldsymbol{x}_j : \boldsymbol{x}_j \in S-H} dist\left(\boldsymbol{x}_j\right)$
    2. Choose $\boldsymbol{h}_i \in S - H$ such that $dist\left(\boldsymbol{h}_i\right) == D$
    3. $H = H \cup \{\boldsymbol{h}_i\}$
    4. for $j = 1$ to N
       * if $L\left(\boldsymbol{x}_j, \boldsymbol{h}_i\right) \leq dist\left(\boldsymbol{x}_j\right)$
         · $dist\left(\boldsymbol{x}_j\right) = L\left(\boldsymbol{x}_j, \boldsymbol{h}_i\right)$
         · $cluster\left(\boldsymbol{x}_j\right) = i$

3.4. K-Medoids

Similar to K-Means, but instead of using the means as centroid some k elements of the data sets. Then, new k centers are chosen to see if they minimize the total distances to the elements:

$$\sum_{i=1}^{N} \sum_{ij=1}^{d} \left|x_{ij} - c_j^i\right| \tag{7}$$

3.5. Fuzzy C-Means

Here, a relaxation of the K-Means allows to obtain a new cost function

$$J_m\left(\mathcal{S}\right) = \sum_{k=1}^{N} \sum_{i=1}^{C} \left[A_i\left(\boldsymbol{x}_k\right)\right]^m \left\|\boldsymbol{x}_k - \boldsymbol{v}_i\right\|^2 \tag{8}$$

Under the constraints:

- $A_i\left(\boldsymbol{x}_k\right) \in [0,1]\, , for 1 \leq k \leq N and 1 \leq i \leq C$
- $\sum_{i=1}^{C} A_i\left(\boldsymbol{x}_k\right) = 1, for 1 \leq k \leq N$
- $0 < \sum_{k=1}^{N} A_i\left(\boldsymbol{x}_k\right) < n, for 1 \leq i \leq C$
- $m > 1$

Fuzzy-C-Mean Algorithm

1. Let $t = 0$.

2. Select an initial fuzzy pseudo-partition.

3. Calculate the initial C cluster centers using,

$$v_i^{(t)} = \frac{\sum_{k=1}^{N} A_i^{(t)}\left(\boldsymbol{x}_k\right)^m \boldsymbol{x}_k}{\sum_{k=1}^{N} A_i^{(t)}\left(\boldsymbol{x}_k\right)^m}. \tag{9}$$

4. Update for each x_k the membership function by

   - Case I: $\left\|\boldsymbol{x}_k - \boldsymbol{v}_i^{(t)}\right\|^2 > 0$ for all $i \in \{1, 2, ..., C\}$ then

$$A_i^{(t+1)}\left(\boldsymbol{x}_k\right) = \frac{1}{\left[\sum_{j=1}^{C}\left\{\frac{\left\|\boldsymbol{x}_k - \boldsymbol{v}_i^{(t)}\right\|^2}{\left\|\boldsymbol{x}_k - \boldsymbol{v}_j^{(t)}\right\|^2}\right\}^{\frac{1}{m-1}}\right]} \tag{10}$$

   - Case II: $\left\|\boldsymbol{x}_k - \boldsymbol{v}_i^{(t)}\right\|^2 = 0$ for some $i \in I \subseteq \{1, 2, ..., C\}$ then define $A_i^{(t+1)}\left(\boldsymbol{x}_k\right)$ by any non-negative number such that $\sum_{i \in I} A_i^{(t+1)}\left(\boldsymbol{x}_k\right) = 1$ and $A_i^{(t+1)}\left(\boldsymbol{x}_k\right) = 0$ for $i \notin I$.

5. If $\left|\mathcal{S}^{(t+1)} - \mathcal{S}^{(t)}\right| = \max_{i,k}\left|A_i^{(t+1)}\left(\boldsymbol{x}_k\right) - A_i^{(t)}\left(\boldsymbol{x}_k\right)\right| \leq \epsilon$ stop; otherwise increase t and go to step 2.